Journal of Applied Mathematics and Physics, 2024, 12, 3315-3327
https://www.scirp.org/journal/jamp
ISSN Online: 2327-4379
ISSN Print: 2327-4352

Game Theory Optimization via Diverse Genetic Crossover Intelligence

David Webb[1], Eric Sandgren[2*]

[1]AdvanSix Inc., Parsippany, New Jersey, USA
[2]Systems Engineering Department, Donaghey College of Engineering & Information Technology, University of Arkansas at Little Rock, Little Rock, Arkansas, USA
Email: david.j.webb@comcast.net, *sandgreneric@gmail.com

How to cite this paper: Webb, D. and Sandgren, E. (2024) Game Theory Optimization via Diverse Genetic Crossover Intelligence. *Journal of Applied Mathematics and Physics*, **12**, 3315-3327.
https://doi.org/10.4236/jamp.2024.1210197

Received: September 7, 2024
Accepted: October 9, 2024
Published: October 12, 2024

Abstract

Game theory is explored via a maze application where combinatorial optimization occurs with the objective of traversing through a defined maze with an aim to enhance decision support and locate the optimal travel sequence while minimizing computation time. This combinatorial optimization approach is initially demonstrated by utilizing a traditional genetic algorithm (GA), followed by the incorporation of artificial intelligence utilizing embedded rules based on domain-specific knowledge. The aim of this initiative is to compare the results of the traditional and rule-based optimization approaches with results acquired through an intelligent crossover methodology. The intelligent crossover approach encompasses a two-dimensional GA encoding where a second chromosome string is introduced within the GA, offering a sophisticated means for chromosome crossover amongst selected parents. Additionally, parent selection intelligence is incorporated where the best-traversed paths or population members are retained and utilized as potential parents to mate with parents selected within a traditional GA methodology. A further enhancement regarding the utilization of saved optimal population members as potential parents is mathematically explored within this literature.

Keywords

Crossover Intelligence, Game Theory, Maze Navigation, Genetic Optimization

1. Introduction

The aim of this initiative is to further enhance solution convergence within a game theory environment via a maze application. This game theory application consists

*Corresponding author; Retired.

of a combinatorial optimization problem where the maze explored requires a series of directional moves, *i.e.* (N, S, E, & W) with the goal of determining the optimal sequence of moves required to navigate and complete the maze. Game theory is further explored within the concept of decision support, as represented by a series of chosen moves within a designated population, which represents travel sequences or decisions to effectively negotiate the maze application. Embedded rules based on domain-specific knowledge are explored within this research, which further aids in the decision-making process when traversing the maze example. As taken directly from Webb, *et al.* [1], one of the most interesting applications of genetic algorithms falls into the area of decision support. Decision support problems involve a series of decisions, each of which is influenced by all decisions made prior to that point. As referenced by Webb, *et al.* [1], decision support has developed into a broad spectrum of applications encompassing optimization through a variety of methods, including genetic algorithms [2] [3], and [4]. A traditional genetic algorithm is an evolutionary approach where problem characteristics are encoded to initially form random chromosome strings where strings are paired, and the exchange of essential data is passed to create offspring. This offspring is evaluated against an objective function and potential optimization constraints, which determine the success of the derived offspring. Additional key factors in the genetic algorithm evolutionary process include penalty factors, which minimize the occurrence of poor offspring and mutation, which randomly alters the encoded string to produce designs potentially unattainable within a small population size. Extensive background and theory regarding the fundamental methodology of the genetic algorithm can be found in [5] and [6].

A traditional GA was utilized as an attempt to solve the maze example; however, this traditional methodology possesses limitations that are heavily dependent on the quality of the initial population, parent selection across generations, and trait or characteristic crossover. The conventional GA becomes limited following a series of generations produced where solution progression becomes solely reliant on mutation. As depicted by Webb, *et al.* [1], traditional GA limitations lacked the ability to successfully negotiate the maze and domain specific knowledge was introduced to further enhance solution convergence. Domain-specific knowledge in the form of rules was employed to guide the GA further toward a successful outcome. As developed and illustrated by Webb, *et al.* [1], domain-specific knowledge enabled the GA to successfully navigate and complete the maze, however, computation time required and the number of paths traversed to complete the maze remain to be further enhanced. Improvements to solution convergence utilizing domain-specific knowledge can be further attained through the implementation of intelligent crossover developed by Webb, *et al.* [7]. Intelligent crossover offers an enhancement in the area of decision support and removes boundaries associated with conventional crossover methods, as referenced in [8] and [9]. This research aims to illustrate the results via a comparison approach between a conventional genetic algorithm and applied intelligent crossover where both applications utilize domain specific knowledge.

2. Background

2.1. Traditional Genetic Optimization

As taken directly from Webb, *et al.* [1], traditional genetic optimization or scenario one began with a series of encoded strings that represent the framework of a maze under investigation. These encoded strings allow the genetic algorithm to determine whenever a move is selected (*i.e.*, N, E, S, W) that the move is a legitimate move within the maze. A maze essentially is a predefined amount of space where a grid or lattice is formed which represents a series of rows and columns of square blocks. Each block formed within the lattice represents a "cell". The surrounding walls of user-specified cells are removed and the formation of a maze becomes apparent. The strings within this input file are four characters long, which represent the directions of north, east, south and west, respectively. Each character within the string can either represent the values of "0" or "1" to appropriately define each cell. A value of "0" for the first character within the first row indicates that there is a north wall within the first cell of the grid. Conversely, a value of "1" for the first character within the first row indicates an absence of a north wall within the first cell of the lattice. Each cell is labeled throughout the maze, where the first block or entrance of the maze is designated by (1, 1), meaning row one and column one, and is located at the lower left corner, as illustrated by the example in Figure 1.

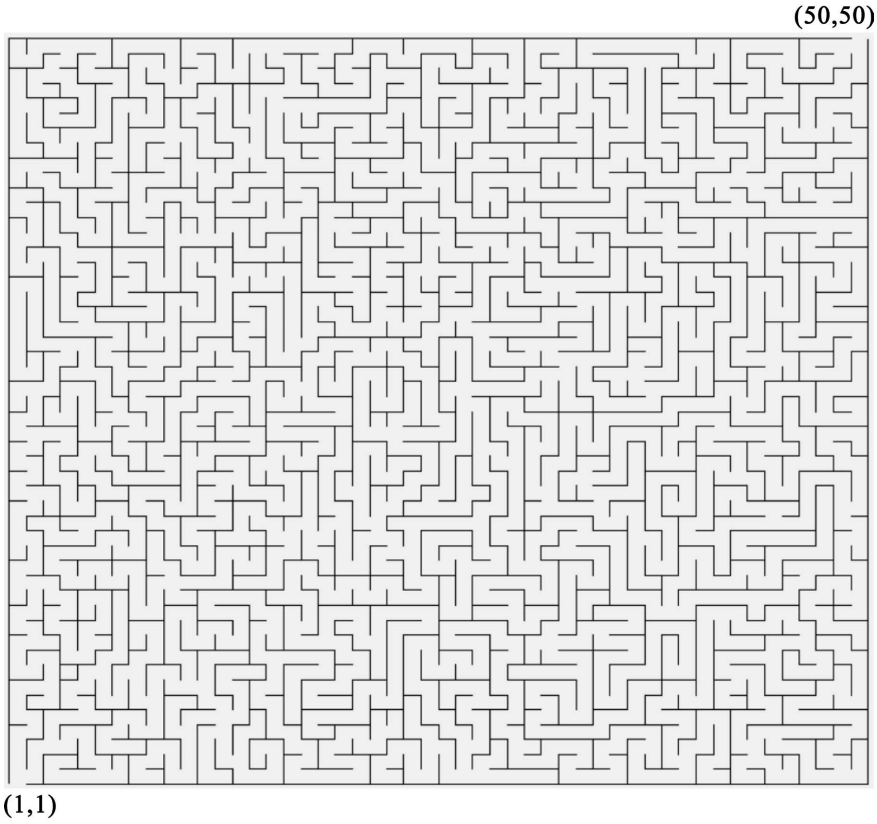

Figure 1. 50 × 50 grid maze.

The exit or last cell for the maze illustrated in Figure 1 is numerically identified by (50, 50), which is the upper right corner of the maze. Strings within the initial maze input file are read from left to right and row by row, beginning with the first cell block. These constant start and end locations were imposed since this generated the longest and most difficult search solution possible for the genetic algorithm. The objective was to locate the travel route which connected the start cell with the end cell. The function of the genetic algorithm was to generate an initial travel sequence string based on travel directions of north, east, south, and west. The number of characters composed of a maximum travel sequence was equal to the number of cells generated by the implementation of a grid. Each character within the travel sequence is assigned a numeric value within a range of four possible values. Table 1 below provides the possible values for which a character within the travel sequence string could potentially represent. Additionally, Table 2 illustrates the genetic algorithm parameters utilized in the problem formulation for all mazes considered.

Table 1. Character representation values for travel sequence string.

Character Value	Representation
1	North
2	East
3	South
4	West

Table 2. Genetic input parameters.

GENETIC INPUT PARAMETERS
Population Size = Maze Grid Size * 10
Probability of Mutation = 0.02
Number of Generations = (5 * Maze Grid Size) + Number of Best Generation Members to Pass to the Next Generation
Penalty Factor = 10
Multiplier of Penalty Factor = 2
Number of Best Generation Members to Pass to the Next Generation = 2
Number of Generations Between Penalty Factor Updates = 10

As taken directly from Webb, *et al.* [1], a random population of travel sequences is generated and evaluated by the objective function as illustrated in Equation (1). If deemed a legitimate move within the maze, the objective function calculates the total distance of each travel sequence; however, the final result is path and distance weighted as illustrated in Equation (2), for the directional moves within the travel sequence are theoretically legitimate, but actual distances may not be calculated in the event N-S or E-W moves are introduced into the travel sequence. Once

evaluated by the objective function, the best travel sequences are selected and placed within a mating pool, where travel sequences are randomly paired and genetic chromosome crossover commences. Chromosome crossover is the process of mating two parent travel sequence strings by the random selection of a numbered character shared by both strings. Once a shared character between both strings is selected, each chromosome string exchanges characteristics to the right of the selected gene until the end of both chromosome strings has been reached. Subsequent to genetic crossover, each offspring produced is evaluated by the objective function, where directional moves within the new travel sequence are only implemented if legitimate within the maze. Illegitimate moves within travel sequences are simply ignored by the objective function and the travel sequence string is reduced in total length. This process continues until a predetermined number of generations of offspring have been produced. Mutation, or the random alteration of genes of a parent travel sequence string within the mating pool, occurs; however, the probability is minimal. Mutation introduces randomness, which provides the design or decision chromosomes with characteristics normally unattainable subsequent to a number of generated offspring. Upon the completion of a user-specified number of generations, the genetic algorithm has potentially created a travel sequence string which encompasses the total or partial path which unites both start and end cells.

Objective function:

$$f(x)=\sqrt{\left(\left(x_{\text{actual}}-x_{\text{final}}\right)^2+\left(y_{\text{actual}}-y_{\text{final}}\right)^2\right)} \tag{1}$$

Weighted objective function: $f(x)_{\text{Weighted}}$

$$f(x)_{\text{Weighted}}=W1-W2*\left(\sqrt{\left(\left(x_{\text{actual}}-x_{\text{final}}\right)+\left(y_{\text{actual}}-y_{\text{final}}\right)^2\right)}-W3*\left(\text{npath}-C\right)\right) \tag{2}$$

where, npath = total number of paths each cell move possesses that culminates the defined path.

x_{actual}, y_{actual}, x_{final}, & y_{final} are parameters that represent the start and end point locations of travel sequences for each maze considered which determine overall travel distance

$W1$, $W2$, & $W3$ are relevant weighting factors to balance out the outcome of the objective function with the number of moves required utilizing constant factor C. The use of a weighted objective function accounts for directional moves within the travel sequence; however, actual distance values may not be calculated in the event N-S or E-W moves are introduced into the travel sequence

2.2. Genetic Optimization via Embedded Rules

To further enhance the decision support concept within game theory, embedded rules were introduced within the maze environment of the traditional GA. These rules were based on domain-specific knowledge introduced by Webb, *et al.* [1]. Travel sequences serve as a series of decisions to render the optimal path sequence,

which ultimately negotiates the maze successfully within a minimal number of iterations or population members evaluated. The aim of this combinatorial optimization problem is to successfully traverse the maze from start to end point locations, as outlined in **Figure 1**, constrained by a multitude of barriers or undesirable travel paths within the defined maze environment. The objective function, as prescribed within Equations (1) and (2), is retained with the incorporation of embedded rules within the traditional GA. As taken directly from Webb, *et al.* [1], domain-specific knowledge in the form of rules were embedded within scenario one. These rules are governed by each travel sequence and the travel sequence is controlled by the genetic algorithm.

The cells that compose the maze are initially predefined by how many exits each cell possesses, and if the cell possesses only one exit, information is provided to which direction the available exit is located (*i.e.*, N, E, S or W). When a cell is located within the travel sequence, and only one exit is available, the genetic code is provided with what direction the cell possesses. With this information, a wall is created to block off the cell, creating a block so that no future travel sequences may enter. All cells are updated with the creation of this new wall, and the process continues whenever a travel sequence encounters a single exit cell. Additionally, when the objective function evaluates each move within a specific travel sequence, redundant moves such as N-S and E-W are removed so that every move accepted creates an actual distance.

The implementation of rules within the objective function subroutine, which eliminates redundant moves before being drawn, is controlled by the genetic code since the travel sequence is governed by the genetic algorithm and rules are applied as travel sequences are introduced to the embedded rules. This also holds true for walls created when travel sequences encounter a cell with only one exit. The rule for the implementation of a wall is only executed if the genetic code generates a travel sequence, which leads to a cell with only one exit. Overall, the formulation of walls upon the location of an undesirable travel sequence that lacks further travel progression eliminates repetitive travel to the undesirable location.

2.3. Intelligent Crossover

The intelligent crossover methodology enhances parent selection and crossover of design traits or characteristics within both global and local arenas. Each parent or design possesses a second encoding string which is utilized to incorporate decision-based intelligence. Secondary encoding strings within selected parents are compared, where differences in comparison value result in the crossover of corresponding traits within the primary encoded or chromosome string. Additionally, to retain an evolutionary approach within the secondary strings, crossover occurs amongst value differences in secondary string parameters. Lastly, parent selection is improved for an intelligent crossover where the incorporation of the best-saved designs or parents are retained and utilized as parents with conventional GA selected parents. To further retain diversity within the parent selection

process, best-saved designs or parents are utilized across odd generations, where the traditional selection of parents is still retained within even generations.

As referenced by Webb and Sandgren [7], the mathematical excerpt below for a 7-digit base 10 problem reveals decision string intelligent crossover interaction, where differences in chromosome value result in the decision for the crossover of characteristics within the primary chromosome string. Additionally, crossover transpires with the secondary chromosome string, where the difference in chromosome string value occurs throughout the string. This intelligent crossover concept is an enhancement within the decision support arena, which is a significant aspect of game theory.

Base 10 crossover intelligence utilizes two chromosome strings where the first string represents a mathematical 7-digit base 10 representation string and the second string represents crossover intelligence to further enhance the solution and convergence timing.

Sample parent one solution string {6, 7, 8, 4, 4, 1, 5} mathematically represents 6,784,415, where the total string length correlates to the optimal 7-digit base 10 solution.

Sample crossover string {2, 2, 1, 2, 1, 1, 2} correlates to parent one sample solution string, where each chromosome value possesses the ability to represent the value of 1 or 2 for crossover instruction. The sample string length corresponds to the total length of the parent one solution string, which provides chromosome crossover instruction for each string member.

Sample parent two solution string {6, 8, 3, 9, 3, 6, 7} mathematically represents 6,839,367 and is the remaining parent selected to mate with the parent one solution string.

Sample crossover string {1, 1, 1, 1, 2, 2, 2} correlates to parent two sample solution string, where each chromosome value possesses the ability to represent the value of 1 or 2 for crossover instruction. The sample string length corresponds to the total length of the parent two solution strings, which provides chromosome crossover instruction for each string member.

Differences between crossover strings result in the crossover of both solution and crossover string characteristics.

Below are the solution and crossover strings that were used prior to intelligent crossover operations.

{6, 7, 8, 4, 4, 1, 5} {2, 2, 1, 2, 1, 1, 2}

{6, 8, 3, 9, 3, 6, 7} {1, 1, 1, 1, 2, 2, 2}

Following intelligent crossover, the offspring solution and crossover strings are below.

{6, 8, 8, 9, 3, 6, 5} {1, 1, 1, 1, 2, 2, 2}

{6, 7, 3, 4, 4, 1, 7} {2, 2, 1, 2, 1, 1, 2}

Key benefits include the selection of parent solution and crossover strings from alternating generations, where parent selection is either from the best solutions

from the prior generation or conventionally selected parent strings for mating.

New developments have been implemented into the parent selection process which are subjected to the intelligent crossover method. Initially, the pool of best-saved parents was arbitrarily determined as the total available best parents for random selection; however, per Equation (3), the following was derived, which serves as the initial pool of best designs or parents for selection.

$$\text{Best}_{\text{Parents}} = \frac{\text{NELEM} + \text{NPOP} + \text{NGEN}_{\text{Total}}}{\frac{\text{NGEN}_{\text{Total}} - \text{NGEN}_{\text{Current}}}{\text{N}_{\text{Choices}}} + \frac{\text{NELEM}}{2}} \tag{3}$$

The following variables are defined below:

NELEM: Number of chromosome string traits per parent

NPOP: Total GA population value

$\text{NGEN}_{\text{Total}}$: Total GA generation value

$\text{NGEN}_{\text{Current}}$: Current generation value within the optimizing GA routine

$\text{N}_{\text{Choices}}$: Number of values each chromosome can possess

An examination of Equation (3) reveals that as the number of generations transpires within the genetic optimization algorithm, the total number or pool of best parents to select increases. This concept allows for generations that have plateaued in value to gain additional best-saved parent(s), which are introduced into the parent selection pool, enabling a continuation of diverse offspring or travel sequences.

3. Results

A 50 × 50 cell maze [10], as depicted in Figure 1, was investigated where comparison results were derived by Webb, *et al.* [1] for the traditional genetic algorithm and coupled with domain-specific knowledge in the form of embedded rules. Results from these previously derived two scenarios are compared with the incorporation of intelligent crossover utilizing the traditional genetic algorithm possessing domain-specific knowledge.

As taken directly from Webb, *et al.* [1], each maze scenario presented was programmed in Visual Basic [11]. Genetic algorithm parameters utilized in the formulation of the provided results are illustrated in Table 3, where function evaluations considered are derived via Equation (4) below.

Table 3. Maze genetic parameters.

Maze Grid Size	Population Size	Generation Size	Function Evaluations
50 × 50	500	252	126,000

$$\text{Function Evaluations} = \text{Population}_{\text{Size}} * \text{Generation}_{\text{Size}} \tag{4}$$

Overall, results depicted within Figures 2-4 demonstrate an increased progression starting from a traditional GA approach, as illustrated in Figure 2. The traditional

GA rendered the inability to locate an optimal solution. Increased performance is depicted in Figure 3, attributed to embedded rules derived from the incorporation of domain-specific knowledge, where the maze was successfully completed within four generations. The incorporation of intelligent crossover further enhanced the embedded rule-based application, as depicted in Figure 4. A comparison between Figure 3 and Figure 4 yields fewer paths traveled, as illustrated by blocked travel routes once the GA reaches a wall or barrier. Intelligent crossover successfully negotiated the maze within the first generation with fewer traveled routes attempted. Crossover intelligence further benefited performance, attributed to a more diverse selection of parents, which yielded an improved pool of candidate offspring for maze route selection. An improvement in parent selection is attributed to the utilization of the best parents evaluated by each prior generation. Equation (3) illustrates an improved selection process of the number of best parents available in the selection pool. Initially, the quantity of best parents available is rounded to 2; however, as the number of generations increases within the optimization routine, the number of best parents available can be rounded to as many as 3. The increase in available best parents assists in the continued progression of diverse offspring as the optimization routine approaches define GA parameters. Lastly, the intelligent crossover methodology enables crossover selection string traits to exchange characteristics, improving the diversity of the crossover intelligence chromosome string.

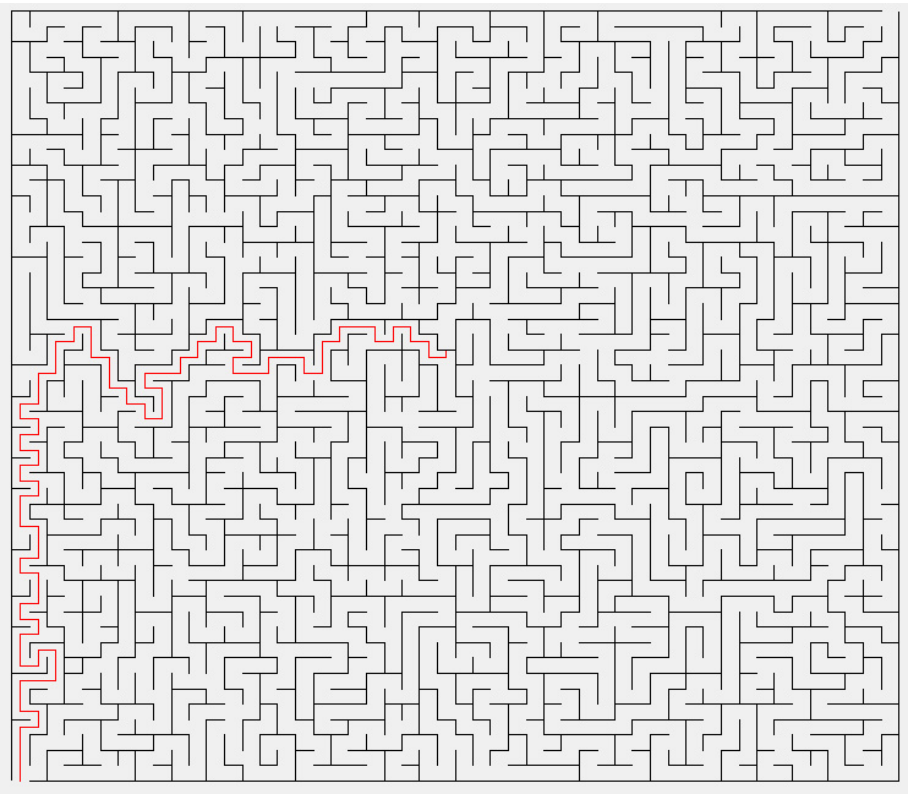

Figure 2. Traditional GA.

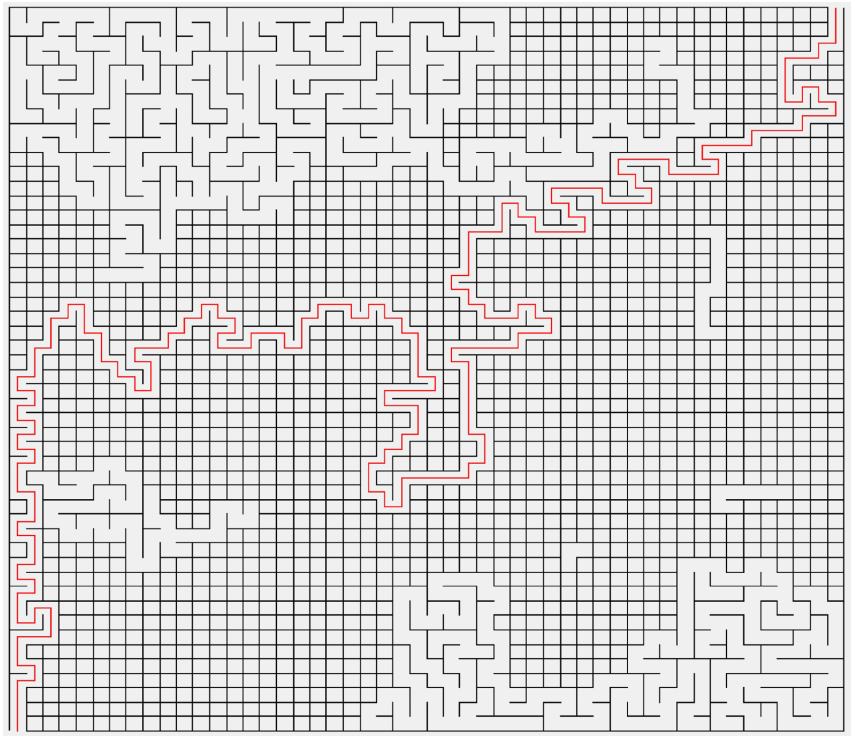

Figure 3. Domain-specific knowledge.

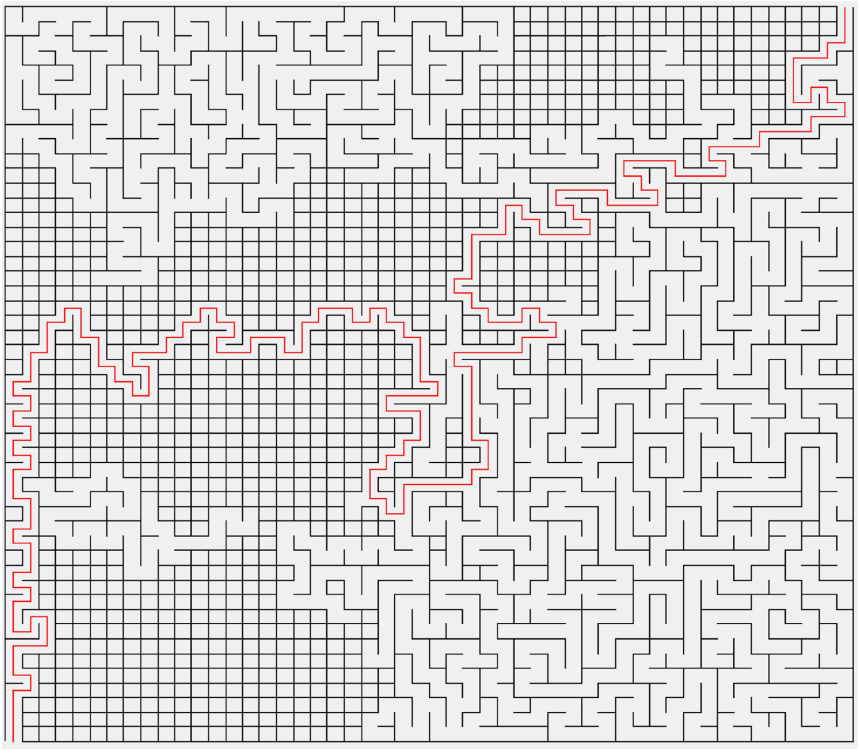

Figure 4. Intelligent crossover methodology.

An examination of scenario one within **Figure 5** revealed a continuous increase in objective function value over a period of one hundred eighty-seven generations

and remained constant in value for the remainder of the specified sixty-five generations. Overall, the traditional GA failed to locate the optimal travel sequence to complete the maze. This is attributed to the lack of diverse offspring or travel sequences to further locate the optimal maze route. Inefficiencies relating to the traditional GA search include a heavy reliance on an evolutionary random search where mutation becomes a key attribute for driving solution convergence.

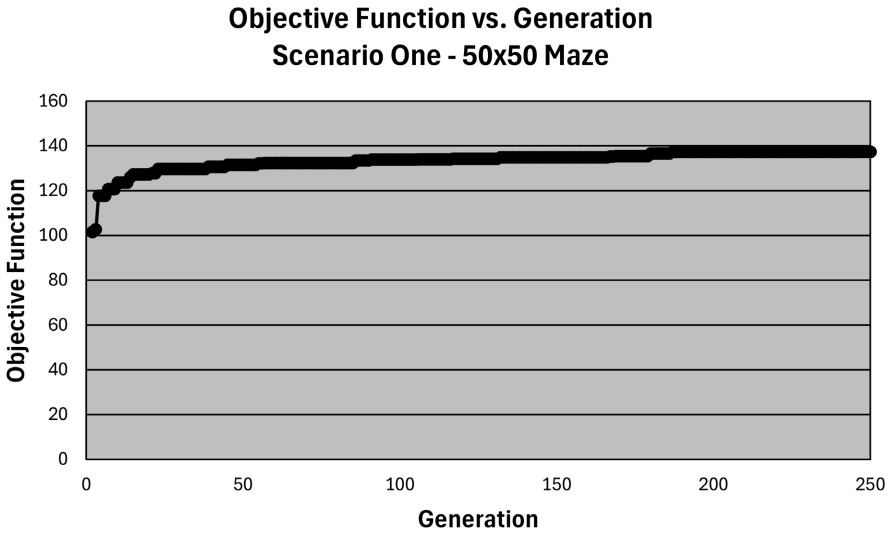

Figure 5. Objective function vs. generation.

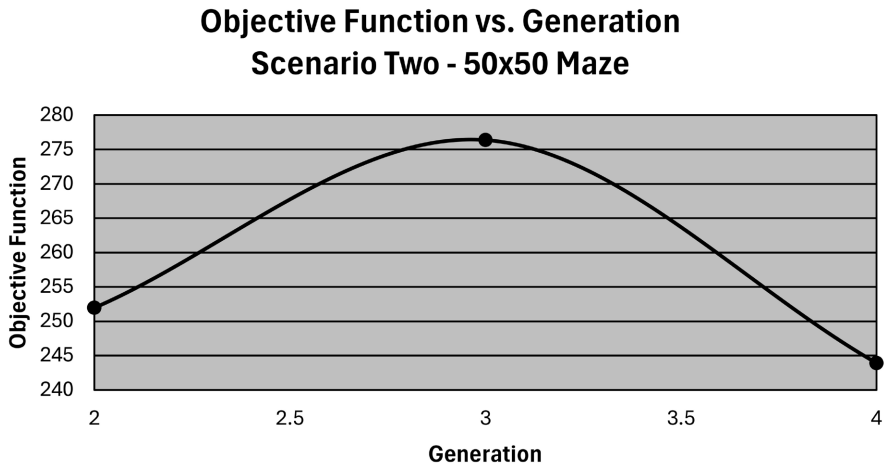

Figure 6. Objective function vs. generation.

Figure 6 depicts scenario two regarding objective function versus generation behavior for a traditional GA possessing embedded rules based on domain-specific knowledge. An aspect of incorporated domain-specific knowledge consisted of the creation of a wall where each travel sequence encountered a barrier, eliminating the GA from revisiting the same total travel sequence more than once. Although proven to drive solution convergence from the traditional GA result in Figure 2, the creation of a barrier or wall prohibits the GA from retaining the best-

saved travel sequence across generations resulting in a fluctuation in objective function behavior as depicted in Figure 6. Since the elimination of paths possessing an attractive objective function value must be trimmed in total path length attributed to the formulation of a wall if a barrier is encountered, it results in a decrease in the objective function value, as illustrated in Figure 6.

Lastly, Figure 7 represents scenario three regarding intelligent crossover as reflected within objective function versus generation behavior. The intelligent crossover methodology diversified offspring, resulting in fewer travel sequences as depicted in Figure 4, which correlates to fewer offspring evaluations, rendering a converged solution within the first generation.

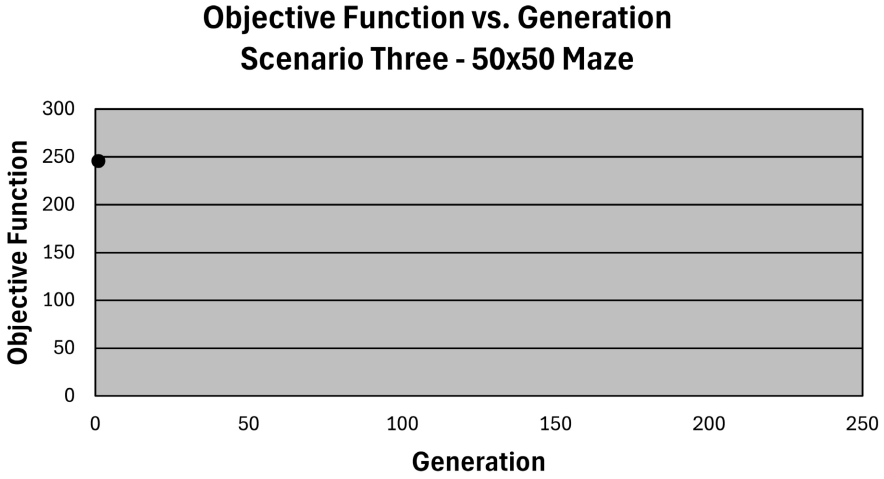

Figure 7. Objective function vs. generation.

4. Concluding Remark

The incorporation of intelligent crossover within a game theory environment enabled further progression of decision support for combinatorial optimization, as depicted within the maze example. Embedded rules within the form of domain-specific knowledge included the elimination of redundant travel sequences (ex. N-S, E-W) coupled with the formulation of walls to eliminate repetitive travel to an undesirable location. The integration of embedded rules resulted in successful maze completion within four generations, as illustrated in Figure 3; however, a reduction in travel routes taken and associated computation time required improvement. The intelligent crossover methodology is not application-specific and can be incorporated into an array of existing GA combinatorial optimization problems. Overall, within the maze application evaluated, crossover intelligence improved GA performance, resulting in the determination of the optimal maze travel sequence within one generation. The ability to diversify offspring and the corresponding intelligence encoding string enables fewer offspring evaluations, as depicted in Figure 4. Additionally, intelligent crossover improves the parent selection abilities of the GA, which includes a pool of the best-evaluated population members. Intelligent crossover offers uniqueness in the selection of a parent based

on the best-evaluated population members coupled with the traditional GA parental selection process. The inclusion of both parent selection avenues creates an effective means for solution convergence, which fosters both local and global solution searches to occur across odd and even generations.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Webb, D.J., Alobaidi, W.M. and Sandgren, E. (2018) Maze Navigation via Genetic Optimization. *Intelligent Information Management*, **10**, 1-15. https://doi.org/10.4236/iim.2018.101001

[2] Alobaidi, W., Sandgren, E. and Alkuam, E. (2017) Decision Support through Intelligent Agent Based Simulation and Multiple Goal Based Evolutionary Optimization. *Intelligent Information Management*, **9**, 97-113. https://doi.org/10.4236/iim.2017.93005

[3] Webb, D., Alobaidi, W. and Sandgren, E. (2017) Structural Design via Genetic Optimization. *Modern Mechanical Engineering*, **7**, 73-90. https://doi.org/10.4236/mme.2017.73006

[4] Alobaidi, W.M., Webb, D.J. and Sandgren, E. (2017) A Rule Based Evolutionary Optimization Approach for the Traveling Salesman Problem. *Intelligent Information Management*, **9**, 115-132. https://doi.org/10.4236/iim.2017.94006

[5] Goldberg, D.E. (1989) Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, Reading MA.

[6] Gen, M. and Cheng, R. (1997) Genetic Algorithms & Engineering Design. Wiley.

[7] Webb, D. and Sandgren, E. (2024) Genetic Optimization via Diverse Crossover Intelligence. *Journal of Applied Mathematics and Physics*, **12**, 2885-2903. https://doi.org/10.4236/jamp.2024.128172

[8] Ahima, M. (2019) A Study of Genetic Algorithm and Crossover Techniques. *International Journal of Computer Science and Mobile Computing*, **8**, 335-344.

[9] Farah Ayiesya, Z., Samad, A. and Fahmi, M.D. (2020) A Review of Crossover Methods and Problem Representation of Genetic Algorithm in Recent Engineering Applications. *International Journal of Advanced Science and Technology*, **29**, 759-769.

[10] Mazes (2004) http://www.mazes.org.uk/

[11] Microsoft Corporation (1998) Microsoft Visual Basic Version 6.0.